THEORY OF COMPUTATION
CSCI 320, course # 7771
Test Solution # 2
May 14, 2012
instructor: Bojana Obrenić

Problem 1 Let:

$$L = \{a^n c^k a^\ell b^j d^m \mid j = k,\ \ell = m,\ n = 0,\ n, k, \ell, j, m \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L. If such a grammar does not exist, prove it.

Answer: Such a grammar does not exist, since L is not context-free.

Observe that the template for L is:

$$c^k a^\ell b^k d^\ell$$

where $k, \ell \geq 0$. Every word of L satisfies all of the following properties.

1. the number of c's is equal to the number of b's;
2. the number of a's is equal to the number of d's.

To prove that L is not context-free, assume the opposite—that L is context-free. Let η be the constant as in the Pumping Lemma for L. Select $k, \ell > \eta$ and consider a word $w = c^k a^\ell b^k d^\ell$. The word w must pump. By the Lemma, in any pumping decomposition, the pumping window cannot extend through more than two of the four segments—it is shorter than η and thereby too short to intersect with three (or more) of them. Then all the cases of acceptable positions of the pumping window and the corresponding pumping instances which violate the stated properties are as follows.

1. the pumping window is entirely within the segment c^k: pump once *up* $\Longrightarrow$ property (1) is lost;
2. the pumping window is entirely within the segment a^ℓ: pump once *up* $\Longrightarrow$ property (2) is lost;
3. the pumping window is entirely within the segment b^k: pump once *up* $\Longrightarrow$ property (1) is lost;
4. the pumping window is entirely within the segment d^ℓ: pump once *up* $\Longrightarrow$ property (2) is lost;

Hence, the Pumping Lemma does not hold for L and L is not context-free.

(b) Draw a state transition graph of a finite automaton that accept L. If such an automaton does not exist, prove it.

Answer: Such a finite automaton does not exist, since L is not regular. To see this, recall that L is not context-free, as is proved in the answer to part (a). If L was regular, it would be context-free (since every regular language is context-free.)

Problem 2 Let:

$$L = \{a^n c^k a^\ell b^j d^m | j = \ell + n, m > 2, k = 0, n, k, \ell, j, m \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L. If such a grammar does not exist, prove it.

Answer: Observe that the template for L is:

$$a^j b^j d^{m+3}$$

where $j, m \geq 0$, whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, D\}$, and the production set P is:

$$\begin{aligned} S &\to ADddd \\ A &\to aAb \mid \lambda \\ D &\to dD \mid \lambda \end{aligned}$$

(b) Draw a state transition graph of a finite automaton that accepts L. If such an automaton does not exist, prove it.

Answer: Such a finite automaton does not exist since L is not regular. To prove this, assume the opposite and observe that all strings of L satisfy the following property:

1. number of a's is equal to the number of b's.

Let η be the constant as in the Pumping Lemma for L. Select a word:

$$w = a^j b^j ddd$$

for some $j > \eta$ (obtained from the general template by setting $m = 0$.) The word w is long enough and must pump. In any pumping decomposition: $w = xyz$, the length of the substring xy is not greater than η, and is thereby less than j. Hence, the pumping window y is located entirely within the segment a^j. Pumping up once produces an excess of a's over b's, violating property (1). Hence, L does not honor the Pumping Lemma and thereby cannot be regular.

Problem 3 Let:

$$L = \{a^n c^k a^\ell b^j d^m \mid k = j,\ n > 2,\ \ell = 0,\ n, k, \ell, j, m \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L. If such a grammar does not exist, prove it.

Answer: Observe that the template for L is:

$$a^{n+3} c^k b^k d^m$$

where $n, k, m \geq 0$, whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{array}{l} S \rightarrow aaaABD \\ A \rightarrow aA \mid \lambda \\ B \rightarrow cBb \mid \lambda \\ D \rightarrow dD \mid \lambda \end{array}$$

(b) Write a regular expression that defines L. If such a regular expression does not exist, prove it.

Answer: Such a regular expression does not exist since L is not regular. To prove that L is not regular, consider first the following two languages:

$$L_1 = \{aaa\, c^k b^k d^m \mid k, m \geq 0\}$$

$$R = aaa\, c^* b^* d^*$$

Observe that R is regular, but also:

$$L_1 = L \cap R$$

Now, if L was regular then its intersection with any regular language would be regular. In particular, L_1 would be regular, since it is the intersection of L with the regular language R. To prove that L is not regular, it is then sufficient to prove that L_1 is not regular.

Next, observe that ever string in the language L_1 satisfies the following property:

1. number of c's is equal to the number of b's;
2. number of a's is equal to 3.

Assume that L_1 is regular, and let η be the constant as in the Pumping Lemma for L. Select a word:

$$w = aaa\, c^k b^k,\ k > \eta$$

The word w is long enough and must pump. In any pumping decomposition: $w = xyz$, the length of the substring xy is not greater than η, which in turn is less than k. Hence, the pumping window y is located entirely within the segment $aaa\, c^k$. However, no part of the pumping window may reside in the segment aaa, since pumping any a's would violate property (2). Then, the pumping window is entirely within the segment c^k, and pumping up once produces an excess of c's, violating property (1). Thus L_1 violates the Pumping Lemma and thereby cannot be regular, meaning that L cannot be regular either.

Problem 4 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, t\}$, $\Sigma = \{a, b, c, d\}$, $\Gamma = \{A, B, D\}$, $F = \{t\}$ and the transition function δ is defined as follows:

$$\begin{array}{l} [q, c, \lambda, t, BAD] \\ [t, a, A, t, \lambda] \\ [t, b, B, t, \lambda] \\ [t, c, \lambda, t, \lambda] \\ [t, d, D, t, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \ldots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 6 distinct strings that belong to L. If this is impossible, state it and explain why.

Advice for Answer: L ia given by the following regular expression.

$$c\, c^* dc^* ac^* bc^*$$

(b) Draw a state transition graph of a finite automaton that accepts L. If such an automaton does not exist, prove it.

Answer: See Figure 1.

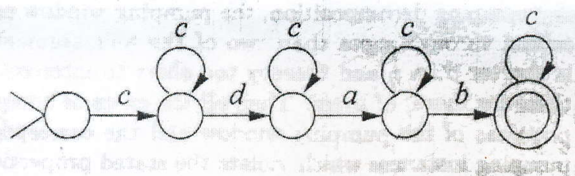

Figure 1:

(c) What is the cardinality of the set L? If it is finite, state the exact number; if it is infinite, state whether it is countable or uncountable.

Answer: L is infinite and countable.

(d) What is the cardinality of the set $\mathcal{P}(L)$ (the set of subsets of L?) If it is finite, state the exact number; if it is infinite, state whether it is countable or uncountable.

Answer: $\mathcal{P}(L)$ is infinite and uncountable.

Problem 5 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, s, t, v\}$, $\Sigma = \{a, b, c, d, e, f\}$, $\Gamma = \{A, B, K, D, E\}$, $F = \{s\}$ and the transition function δ is defined as follows:

$$\begin{array}{ll} [q, a, \lambda, q, A] & [s, a, A, s, \lambda] \\ [q, a, \lambda, q, E] & [s, b, B, s, \lambda] \\ [q, b, \lambda, q, B] & [s, c, K, s, \lambda] \\ [q, b, \lambda, q, K] & [s, d, D, t, \lambda] \\ [q, d, \lambda, q, D] & [s, e, E, s, \lambda] \\ [q, \lambda, \lambda, s, \lambda] & [t, f, \lambda, v, \lambda] \\ & [v, f, \lambda, s, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 6 distinct strings that belong to L. If this is impossible, state it and explain why.

Answer:

$$aa, ae, abbe, abddffca, babaecab, dadbcdffedff$$

(b) Write a complete formal definition of a context-free grammar that generates L. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e, f\}$, $V = \{S\}$, and the production set P is:

$$S \to aSa \mid aSe \mid bSb \mid bSc \mid dSdff \mid \lambda$$

(c) State one trivial property of the language L, such that a^*b^* does not have this property. Explain carefully why this property is trivial, and prove that L indeed has it, while a^*b^* does not. If such a property does not exist, state it, and explain why it is so.

Answer: Such a property does not exist—if L had this property but a^*b^* did not have it, then the property would by definition be non-trivial.

(d) State one non-trivial property of the language L, such that a^*b^* does not have this property. Explain carefully why this property is non-trivial, and prove that L indeed has it, while a^*b^* does not. If such a property does not exist, state it, and explain why it is so.

Answer: One of the infinitely many proerties which answer to these requirements is:

contains at least one word which contains c.

The property is non-trivial, since at least one language, in this case L, has this property while at least one language, in this case a^*b^*, does not have this proeprty. To see that the property is true for L, inspect the grammar constructed in the part (b). To see that it is false for a^*b^*, observe that all strings of the latter contain no letters other than a and b.

Problem 6 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, r, t\}$, $\Sigma = \{a, b, c\}$, $\Gamma = \{B\}$, $F = \{t\}$ and the transition function δ is defined as follows:

$$\begin{array}{ll} [q, a, \lambda, r, B] & [r, b, B, r, \lambda] \\ [q, a, \lambda, r, BB] & [r, c, \lambda, t, \lambda] \\ [q, a, \lambda, r, BBB] & [t, c, \lambda, t, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 6 distinct strings that belong to L. If this is impossible, state it and explain why.

Advice for Answer: L ia given by the following regular expression.

$$a\ (b \cup bb \cup bbb)\ c c^*$$

(b) Write a complete formal definition of a context-free grammar that generates L. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, K\}$, and the production set P is:

$$\begin{aligned} S &\to aBcK \\ B &\to b \mid bb \mid bb \\ K &\to cK \mid \lambda \end{aligned}$$

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w.

OUTPUT: yes if w is an element of the language accepted by the pushdown automaton M (defined at the beginning of this problem);
no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist, and prove it.

Answer: Convert the regular expression given in the answer to part (a) into a finite automaton, convert this automaton into its deterministic equivalent, and simulate it.

Problem 7 Consider the following Turing machine: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $\Sigma = \{a, b, c\}$; $Q = \{q, r, s, p, v, t, z, x, y, m\}$; $\Gamma = \{B, a, b, c\}$; $F = \{x\}$; and δ is defined by the following transition set:

$$
\begin{array}{ll}
[q, a, r, a, R] & [v, a, z, a, L] \\
 & [v, b, y, b, L] \\
[r, b, s, b, R] & [v, c, m, c, L] \\
 & \\
[s, c, t, c, R] & [z, a, x, a, L] \\
 & [z, b, y, b, L] \\
[t, a, p, a, R] & [z, c, m, c, L] \\
[t, b, p, b, R] & \\
[t, c, p, c, R] & [y, a, y, a, R] \\
 & [y, b, y, b, R] \\
[p, a, p, a, R] & [y, c, y, c, R] \\
[p, b, p, b, R] & [y, B, y, B, R] \\
[p, c, p, c, R] & \\
[p, B, v, B, L] &
\end{array}
$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L be the set of strings on which M *diverges*.

(a) List 6 distinct strings that belong to L. If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (b).

(b) Write a regular expression that defines L. If such a regular expression does not exist, prove it.

Answer:

$$abc\ (a \cup b \cup c)^*\ (b \cup ba)$$

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w over $\{a, b, c\}$.

OUTPUT: yes if w is a string that represents a Turing Machine which accepts exactly those strings that do not belong to the set L (defined at the beginning of this problem);
no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist, and prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing Machines whose languages have the following property:

is equal to $\overline{L}$.

This property is non-trivial because (by construction) the language $\overline{L}$ has it, while the language L does not have it. Hence, by Rice's Theorem the described algorithm is impossible.

Problem 8 Consider the following Turing machine: $M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, p, s, t, v\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{1, 0, B\}$; and δ is defined by the following transition set:

$$
\begin{array}{ll}
[q, 0, q, 0, R] & [t, 0, v, 0, L] \\
[q, 1, p, 1, R] & [t, 1, s, 1, R] \\
[q, B, q, B, R] & \\
 & [v, 0, v, 0, L] \\
[p, 0, p, 0, R] & [v, 1, s, 1, R] \\
[p, 1, t, 1, L] & \\
[p, B, p, B, R] & [s, 0, s, 0, R] \\
 & [s, 1, s, 1, R] \\
 & [s, B, s, B, R]
\end{array}
$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right only.) B is the designated blank symbol.)

Let L be the set of string on which M halts.

(a) List 6 distinct strings that belong to L. If this is impossible, state it and explain why.

Answer: Impossible, because M does not halt on any input string.

(b) Write a regular expression that defines L. If such a regular expression does not exist, prove it.

Answer:

$$\emptyset$$

Advice for Answer: Straightforwardly, M diverges unless it finds the second occurrence of 1. If it finds the second occurrence of 1, then it skips over all the 0's (if any) to the left of this second occurrence of 1 until it reaches (as it must) the first occurrence of 1, at which point it escapes to diverge.

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w over $\{0, 1\}$.

OUTPUT: yes if w is an element of the set of exactly those strings on which the Turing Machine M (defined at the beginning of this problem) diverges;
no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist and prove it.

Answer: Return yes.

Problem 9 Consider the following Turing machine: $M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, s, x, t, p, y, z, v.m\}$; $\Sigma = \{0,1\}$; $\Gamma = \{B, 0, 1\}$; $F = \{m\}$;
and δ is defined by the following transition set:

$[q, 0, q, 0, R]$	$[y, 1, z, 1, L]$
$[q, 1, s, 1, R]$	$[y, 0, x, 0, R]$
$[q, B, x, B, R]$	
	$[z, 1, x, 1, R]$
$[s, 0, q, 0, R]$	$[z, 0, v, 0, R]$
$[s, 1, t, 1, L]$	
$[s, B, x, B, R]$	$[v, 1, x, 1, R]$
	$[v, 0, m, 0, R]$
$[t, 0, q, 0, R]$	
$[t, 1, p, 1, R]$	$[x, 0, x, 0, R]$
$[t, B, x, B, R]$	$[x, 1, x, 1, R]$
	$[x, B, x, B, R]$
$[p, 0, p, 0, R]$	
$[p, 1, p, 1, R]$	
$[p, B, y, B, L]$	

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right only.) M accepts by final state. B is the designated blank symbol.)

Let L be the set of string which M accepts.

(a) List 6 distinct strings that belong to L. If this is impossible, state it and explain why.

Answer: Impossible, because M does not accept any input string.

(b) Write a regular expression that defines L. If such a regular expression does not exist, prove it.

Answer:

$$\emptyset$$

Advice for Answer: Straightforwardly, M diverges unless it finds two consecutive occurrences of 1. If it finds two consecutive occurrences of 1, then it returns to the left to examine the first of them (in state t) which then leads to skipping over the entire input (moving to the right in state p) and inspection of the last input symbol (in state y.) If the last symbol is 0, M escapes to diverge. If the last symbol is 1, M inspects the next-to-last symbol (in state z.) If the next-to-last symbol is 1, it escapes to diverge. If the next-to-last symbol is 0, it returns to inspect the last symbol again (in state v.) Since the last symbol (in this case) is always 1, it escapes to diverge.

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w over $\{0, 1\}$.

OUTPUT: yes if w is a string such that the Turing Machine represented by w halts exactly when the machine M (defined at the beginning of this problem) accepts;
no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist and prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing Machines whose languages have the following property:

is empty.

This property is non-trivial because (by construction) the language $L(M)$ has it, while the language $L(M)$ does not have it. Hence, by Rice's Theorem the described algorithm is impossible.

Problem 10 Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the string begins with a, then both of the following conditions hold:
   (a) the string does not end with a;
   (b) if the last symbol of the string was altered so that it becomes a, then the resulting string would be a palindrome;
2. if the the string begins with b, then both of the following conditions hold:
   (a) the string has an odd length;
   (b) the middle symbol is equal to the last symbol;
3. if the string begins with c, then its length is divisible by 3.

Write a complete formal definition of a context-free grammar that generates the language L. If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where
$\Sigma = \{a, b, c\}$, $V = \{S, A, B, K, P, E, F, H, Z, K, L\}$, and the production set P is:

$$S \rightarrow A \mid B \mid K$$

$$A \rightarrow aPb \mid aPc$$
$$P \rightarrow aPa \mid bPb \mid cPc \mid Z \mid \lambda$$

$$B \rightarrow bEa \mid bFb \mid bHc$$
$$E \rightarrow ZEZ \mid a$$
$$F \rightarrow ZFZ \mid b$$
$$H \rightarrow ZHZ \mid c$$

$$K \rightarrow cZZL$$
$$L \rightarrow ZZZL \mid \lambda$$

$$Z \rightarrow a \mid b \mid c$$